UNIVERSIDAD REY JUAN CARLOS

TRABAJO FIN DE GRADO

Aplicación para la compartición segura de ficheros

Autor:
Sergio Merino Hernández

Tutor:
Dr. Gorka Guardiola Múzquiz

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN
GRADO EN INGENIERÍA EN TELEMÁTICA

16 de noviembre de 2017

Resumen

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too…

Agradecimientos

The acknowledgments and the people to thank go here, don't forget to include your project advisor...

Índice general

Índice de figuras

Índice de cuadros

Lista de Abreviaciones

AES	**A**dvanced **E**ncryption **S**tandard
API	**A**pplication **P**rogramming **I**nterface
ART	**A**ndroid **R**un**t**ime
BC	**B**ouncy **C**astle
CBC	**C**ipher **B**lock **C**haining
CFB	**C**ipher **F**eed**b**ack
ECB	**E**lectronic **C**ode**b**ook
HTTP	**H**yper**t**ext **T**ransfer **P**rotocol
IV	**I**nitialization **V**ector
JCA	**J**ava **C**ryptography **A**rchitecture
JCE	**J**ava **C**ryptography **E**xtension
JVM	**J**ava **V**irtual **M**achine
MAC	**M**essage **A**uthentication **C**ode
MCD	**M**áximo **C**omún **D**ivisor
mcm	**m**ínimo **c**omún **m**últiplo
OFB	**O**utput **F**eed**b**ack
PSS	**P**robabilistic **S**ignature **S**cheme
RSA	**R**ivest - **S**hamir - **A**dleman
SSA	**S**ignature **S**cheme with **A**ppendix

For/Dedicated to/To my…

Capítulo 1

Uso de la plantilla

1.1. Welcome and Thank You

Welcome to this LaTeX Thesis Template, a beautiful and easy to use template for writing a thesis using the LaTeX typesetting system.

If you are writing a thesis (or will be in the future) and its subject is technical or mathematical (though it doesn't have to be), then creating it in LaTeX is highly recommended as a way to make sure you can just get down to the essential writing without having to worry over formatting or wasting time arguing with your word processor.

LaTeX is easily able to professionally typeset documents that run to hundreds or thousands of pages long. With simple mark-up commands, it automatically sets out the table of contents, margins, page headers and footers and keeps the formatting consistent and beautiful. One of its main strengths is the way it can easily typeset mathematics, even *heavy* mathematics. Even if those equations are the most horribly twisted and most difficult mathematical problems that can only be solved on a super-computer, you can at least count on LaTeX to make them look stunning.

1.2. Learning LaTeX

LaTeX is not a WYSIWYG (What You See is What You Get) program, unlike word processors such as Microsoft Word or Apple's Pages. Instead, a document written for LaTeX is actually a simple, plain text file that contains *no formatting*. You tell LaTeX how you want the formatting in the finished document by writing in simple commands amongst the text, for example, if I want to use *italic text for emphasis*, I write the `\emph{text}` command and put the text I want in italics in between the curly braces. This means that LaTeX is a «mark-up» language, very much like HTML.

1.2.1. A (not so short) Introduction to LaTeX

If you are new to LaTeX, there is a very good eBook – freely available online as a PDF file – called, «The Not So Short Introduction to LaTeX». The book's title is typically shortened to just *lshort*. You can download the latest version (as it is occasionally updated) from here: `http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf`

It is also available in several other languages. Find yours from the list on this page: `http://www.ctan.org/tex-archive/info/lshort/`

It is recommended to take a little time out to learn how to use LaTeX by creating several, small 'test' documents, or having a close look at several templates on: `http://www.LaTeXTemplates.com`

Making the effort now means you're not stuck learning the system when what you *really* need to be doing is writing your thesis.

1.2.2. A Short Math Guide for LaTeX

If you are writing a technical or mathematical thesis, then you may want to read the document by the AMS (American Mathematical Society) called, «A Short Math Guide for LaTeX». It can be found online here: `http://www.ams.org/tex/amslatex.html` under the «Additional Documentation» section towards the bottom of the page.

1.2.3. Common LaTeX Math Symbols

There are a multitude of mathematical symbols available for LaTeX and it would take a great effort to learn the commands for them all. The most common ones you are likely to use are shown on this page: `http://www.sunilpatel.co.uk/latex-type/latex-math-symbols/`

You can use this page as a reference or crib sheet, the symbols are rendered as large, high quality images so you can quickly find the LaTeX command for the symbol you need.

1.2.4. LaTeX on a Mac

The LaTeX distribution is available for many systems including Windows, Linux and Mac OS X. The package for OS X is called MacTeX and it contains all the applications you need – bundled together and pre-customized – for a fully working LaTeX environment and work flow.

MacTeX includes a custom dedicated LaTeX editor called TeXShop for writing your '`.tex`' files and BibDesk: a program to manage your references and create your bibliography section just as easily as managing songs and creating playlists in iTunes.

1.3. Getting Started with this Template

If you are familiar with LaTeX, then you should explore the directory structure of the template and then proceed to place your own information into the *THESIS INFORMATION* block of the `main.tex` file. You can then modify the rest of this file to your unique specifications based on your degree/university. Section 1.5 on page 4 will help you do this. Make sure you also read section 1.7 about thesis conventions to get the most out of this template.

If you are new to LaTeX it is recommended that you carry on reading through the rest of the information in this document.

Before you begin using this template you should ensure that its style complies with the thesis style guidelines imposed by your institution. In most cases this template style and layout will be suitable. If it is not, it may only require a small change to bring the template in line with your institution's recommendations. These modifications will need to be done on the `MastersDoctoralThesis.cls` file.

1.3.1. About this Template

This LaTeX Thesis Template is originally based and created around a LaTeX style file created by Steve R. Gunn from the University of Southampton (UK), department of Electronics and Computer Science. You can find his original thesis style file at his site, here: `http://www.ecs.soton.ac.uk/~srg/softwaretools/document/templates/`

Steve's `ecsthesis.cls` was then taken by Sunil Patel who modified it by creating a skeleton framework and folder structure to place the thesis files in. The resulting template can be found on Sunil's site here: `http://www.sunilpatel.co.uk/thesis-template`

Sunil's template was made available through `http://www.LaTeXTemplates.com` where it was modified many times based on user requests and questions. Version 2.0 and onwards of this template represents a major modification to Sunil's template and is, in fact, hardly recognisable. The work to make version 2.0 possible was carried out by Vel and Johannes Böttcher.

1.4. What this Template Includes

1.4.1. Folders

This template comes as a single zip file that expands out to several files and folders. The folder names are mostly self-explanatory:

Appendices – this is the folder where you put the appendices. Each appendix should go into its own separate `.tex` file. An example and template are included in the directory.

Chapters – this is the folder where you put the thesis chapters. A thesis usually has about six chapters, though there is no hard rule on this. Each chapter should go in its own separate `.tex` file and they can be split as:

- Chapter 1: Introduction to the thesis topic
- Chapter 2: Background information and theory
- Chapter 3: (Laboratory) experimental setup
- Chapter 4: Details of experiment 1
- Chapter 5: Details of experiment 2
- Chapter 6: Discussion of the experimental results
- Chapter 7: Conclusion and future directions

This chapter layout is specialised for the experimental sciences, your discipline may be different.

Figures – this folder contains all figures for the thesis. These are the final images that will go into the thesis document.

1.4.2. Files

Included are also several files, most of them are plain text and you can see their contents in a text editor. After initial compilation, you will see that more auxiliary

files are created by LaTeX or BibTeX and which you don't need to delete or worry about:

example.bib – this is an important file that contains all the bibliographic information and references that you will be citing in the thesis for use with BibTeX. You can write it manually, but there are reference manager programs available that will create and manage it for you. Bibliographies in LaTeX are a large subject and you may need to read about BibTeX before starting with this. Many modern reference managers will allow you to export your references in BibTeX format which greatly eases the amount of work you have to do.

MastersDoctoralThesis.cls – this is an important file. It is the class file that tells LaTeX how to format the thesis.

main.pdf – this is your beautifully typeset thesis (in the PDF file format) created by LaTeX. It is supplied in the PDF with the template and after you compile the template you should get an identical version.

main.tex – this is an important file. This is the file that you tell LaTeX to compile to produce your thesis as a PDF file. It contains the framework and constructs that tell LaTeX how to layout the thesis. It is heavily commented so you can read exactly what each line of code does and why it is there. After you put your own information into the *THESIS INFORMATION* block – you have now started your thesis!

Files that are *not* included, but are created by LaTeX as auxiliary files include:

main.aux – this is an auxiliary file generated by LaTeX, if it is deleted LaTeX simply regenerates it when you run the main `.tex` file.

main.bbl – this is an auxiliary file generated by BibTeX, if it is deleted, BibTeX simply regenerates it when you run the `main.aux` file. Whereas the `.bib` file contains all the references you have, this `.bbl` file contains the references you have actually cited in the thesis and is used to build the bibliography section of the thesis.

main.blg – this is an auxiliary file generated by BibTeX, if it is deleted BibTeX simply regenerates it when you run the main `.aux` file.

main.lof – this is an auxiliary file generated by LaTeX, if it is deleted LaTeX simply regenerates it when you run the main `.tex` file. It tells LaTeX how to build the *List of Figures* section.

main.log – this is an auxiliary file generated by LaTeX, if it is deleted LaTeX simply regenerates it when you run the main `.tex` file. It contains messages from LaTeX, if you receive errors and warnings from LaTeX, they will be in this `.log` file.

main.lot – this is an auxiliary file generated by LaTeX, if it is deleted LaTeX simply regenerates it when you run the main `.tex` file. It tells LaTeX how to build the *List of Tables* section.

main.out – this is an auxiliary file generated by LaTeX, if it is deleted LaTeX simply regenerates it when you run the main `.tex` file.

So from this long list, only the files with the `.bib`, `.cls` and `.tex` extensions are the most important ones. The other auxiliary files can be ignored or deleted as LaTeX and BibTeX will regenerate them.

1.5. Filling in Your Information in the `main.tex` File

You will need to personalise the thesis template and make it your own by filling in your own information. This is done by editing the `main.tex` file in a text editor or your favourite LaTeX environment.

Open the file and scroll down to the third large block titled *THESIS INFORMATION* where you can see the entries for *University Name*, *Department Name*, etc ...

Fill out the information about yourself, your group and institution. You can also insert web links, if you do, make sure you use the full URL, including the `http://` for this. If you don't want these to be linked, simply remove the `\href{url}{name}` and only leave the name.

When you have done this, save the file and recompile `main.tex`. All the information you filled in should now be in the PDF, complete with web links. You can now begin your thesis proper!

1.6. The `main.tex` File Explained

The `main.tex` file contains the structure of the thesis. There are plenty of written comments that explain what pages, sections and formatting the LaTeX code is creating. Each major document element is divided into commented blocks with titles in all capitals to make it obvious what the following bit of code is doing. Initially there seems to be a lot of LaTeX code, but this is all formatting, and it has all been taken care of so you don't have to do it.

Begin by checking that your information on the title page is correct. For the thesis declaration, your institution may insist on something different than the text given. If this is the case, just replace what you see with what is required in the *DECLARATION PAGE* block.

Then comes a page which contains a funny quote. You can put your own, or quote your favourite scientist, author, person, and so on. Make sure to put the name of the person who you took the quote from.

Following this is the abstract page which summarises your work in a condensed way and can almost be used as a standalone document to describe what you have done. The text you write will cause the heading to move up so don't worry about running out of space.

Next come the acknowledgements. On this page, write about all the people who you wish to thank (not forgetting parents, partners and your advisor/supervisor).

The contents pages, list of figures and tables are all taken care of for you and do not need to be manually created or edited. The next set of pages are more likely to be optional and can be deleted since they are for a more technical thesis: insert a list of abbreviations you have used in the thesis, then a list of the physical constants and numbers you refer to and finally, a list of mathematical symbols used in any formulae. Making the effort to fill these tables means the reader has a one-stop place to refer to instead of searching the internet and references to try and find out what you meant by certain abbreviations or symbols.

The list of symbols is split into the Roman and Greek alphabets. Whereas the abbreviations and symbols ought to be listed in alphabetical order (and this is *not* done automatically for you) the list of physical constants should be grouped into similar themes.

The next page contains a one line dedication. Who will you dedicate your thesis to?

Finally, there is the block where the chapters are included. Uncomment the lines (delete the % character) as you write the chapters. Each chapter should be written in its own file and put into the *Chapters* folder and named `Chapter1`, `Chapter2`, etc... Similarly for the appendices, uncomment the lines as you need them. Each appendix should go into its own file and placed in the *Appendices* folder.

After the preamble, chapters and appendices finally comes the bibliography. The bibliography style (called *authoryear*) is used for the bibliography and is a fully featured style that will even include links to where the referenced paper can be found online. Do not underestimate how grateful your reader will be to find that a reference to a paper is just a click away. Of course, this relies on you putting the URL information into the BibTeX file in the first place.

1.7. Thesis Features and Conventions

To get the best out of this template, there are a few conventions that you may want to follow.

One of the most important (and most difficult) things to keep track of in such a long document as a thesis is consistency. Using certain conventions and ways of doing things (such as using a Todo list) makes the job easier. Of course, all of these are optional and you can adopt your own method.

1.7.1. Printing Format

This thesis template is designed for double sided printing (i.e. content on the front and back of pages) as most theses are printed and bound this way. Switching to one sided printing is as simple as uncommenting the *oneside* option of the `documentclass` command at the top of the `main.tex` file. You may then wish to adjust the margins to suit specifications from your institution.

The headers for the pages contain the page number on the outer side (so it is easy to flick through to the page you want) and the chapter name on the inner side.

The text is set to 11 point by default with single line spacing, again, you can tune the text size and spacing should you want or need to using the options at the very start of `main.tex`. The spacing can be changed similarly by replacing the *singlespacing* with *onehalfspacing* or *doublespacing*.

1.7.2. Using US Letter Paper

The paper size used in the template is A4, which is the standard size in Europe. If you are using this thesis template elsewhere and particularly in the United States, then you may have to change the A4 paper size to the US Letter size. This can be done in the margins settings section in `main.tex`.

Due to the differences in the paper size, the resulting margins may be different to what you like or require (as it is common for institutions to dictate certain margin sizes). If this is the case, then the margin sizes can be tweaked by modifying the values in the same block as where you set the paper size. Now your document should be set up for US Letter paper size with suitable margins.

1.7.3. References

The `biblatex` package is used to format the bibliography and inserts references such as this one (). The options used in the `main.tex` file mean that the in-text citations of references are formatted with the author(s) listed with the date of the publication. Multiple references are separated by semicolons (e.g. ()) and references with more than three authors only show the first author with *et al.* indicating there are more authors (e.g. ()). This is done automatically for you. To see how you use

references, have a look at the `Chapter1.tex` source file. Many reference managers allow you to simply drag the reference into the document as you type.

Scientific references should come *before* the punctuation mark if there is one (such as a comma or period). The same goes for footnotes[1]. You can change this but the most important thing is to keep the convention consistent throughout the thesis. Footnotes themselves should be full, descriptive sentences (beginning with a capital letter and ending with a full stop). The APA6 states: «Footnote numbers should be superscripted, [...], following any punctuation mark except a dash.» The Chicago manual of style states: «A note number should be placed at the end of a sentence or clause. The number follows any punctuation mark except the dash, which it precedes. It follows a closing parenthesis.»

The bibliography is typeset with references listed in alphabetical order by the first author's last name. This is similar to the APA referencing style. To see how LaTeX typesets the bibliography, have a look at the very end of this document (or just click on the reference number links in in-text citations).

A Note on bibtex

The bibtex backend used in the template by default does not correctly handle unicode character encoding (i.e. ïnternationalçharacters). You may see a warning about this in the compilation log and, if your references contain unicode characters, they may not show up correctly or at all. The solution to this is to use the biber backend instead of the outdated bibtex backend. This is done by finding this in `main.tex`: *`backend=bibtex`* and changing it to *`backend=biber`*. You will then need to delete all auxiliary BibTeX files and navigate to the template directory in your terminal (command prompt). Once there, simply type `biber main` and biber will compile your bibliography. You can then compile `main.tex` as normal and your bibliography will be updated. An alternative is to set up your LaTeX editor to compile with biber instead of bibtex, see here for how to do this for various editors.

1.7.4. Tables

Tables are an important way of displaying your results, below is an example table which was generated with this code:

```
\begin{table}
\caption{The effects of treatments X and Y on the four groups studied.}
\label{tab:treatments}
\centering
\begin{tabular}{l l l}
\toprule
\tabhead{Groups} & \tabhead{Treatment X} & \tabhead{Treatment Y} \\
\midrule
1 & 0.2 & 0.8\\
2 & 0.17 & 0.7\\
3 & 0.24 & 0.75\\
4 & 0.68 & 0.3\\
\bottomrule\\
\end{tabular}
\end{table}
```

[1] Such as this footnote, here down at the bottom of the page.

CUADRO 1.1: The effects of treatments X and Y on the four groups studied.

Groups	Treatment X	Treatment Y
1	0.2	0.8
2	0.17	0.7
3	0.24	0.75
4	0.68	0.3

You can reference tables with `\ref{<label>}` where the label is defined within the table environment. See `Chapter1.tex` for an example of the label and citation (e.g. Table 1.1).

1.7.5. Figures

There will hopefully be many figures in your thesis (that should be placed in the *Figures* folder). The way to insert figures into your thesis is to use a code template like this:

```
\begin{figure}
\centering
\includegraphics{Figures/Electron}
\decoRule
\caption[An Electron]{An electron (artist's impression).}
\label{fig:Electron}
\end{figure}
```

Also look in the source file. Putting this code into the source file produces the picture of the electron that you can see in the figure below.

Sometimes figures don't always appear where you write them in the source. The placement depends on how much space there is on the page for the figure. Sometimes there is not enough room to fit a figure directly where it should go (in relation to the text) and so LaTeX puts it at the top of the next page. Positioning figures is the job of LaTeX and so you should only worry about making them look good!

Figures usually should have captions just in case you need to refer to them (such as in Figure 1.1). The `\caption` command contains two parts, the first part, inside the square brackets is the title that will appear in the *List of Figures*, and so should be short. The second part in the curly brackets should contain the longer and more descriptive caption text.

The `\decoRule` command is optional and simply puts an aesthetic horizontal line below the image. If you do this for one image, do it for all of them.

LaTeX is capable of using images in pdf, jpg and png format.

1.7.6. Typesetting mathematics

If your thesis is going to contain heavy mathematical content, be sure that LaTeX will make it look beautiful, even though it won't be able to solve the equations for you.

The «Not So Short Introduction to LaTeX» (available on CTAN) should tell you everything you need to know for most cases of typesetting mathematics. If you need

FIGURA 1.1: An electron (artist's impression).

more information, a much more thorough mathematical guide is available from the AMS called, «A Short Math Guide to LaTeX» and can be downloaded from: `ftp://ftp.ams.org/pub/tex/doc/amsmath/short-math-guide.pdf`

There are many different LaTeX symbols to remember, luckily you can find the most common symbols in The Comprehensive LaTeX~Symbol List.

You can write an equation, which is automatically given an equation number by LaTeX like this:

```
\begin{equation}
E = mc^{2}
\label{eqn:Einstein}
\end{equation}
```

This will produce Einstein's famous energy-matter equivalence equation:

$$E = mc^2 \tag{1.1}$$

All equations you write (which are not in the middle of paragraph text) are automatically given equation numbers by LaTeX. If you don't want a particular equation numbered, use the unnumbered form:

```
\[ a^{2}=4 \]
```

1.8. Sectioning and Subsectioning

You should break your thesis up into nice, bite-sized sections and subsections. LaTeX automatically builds a table of Contents by looking at all the `\chapter{}`, `\section{}` and `\subsection{}` commands you write in the source.

The Table of Contents should only list the sections to three (3) levels. A `chapter{}` is level zero (0). A `\section{}` is level one (1) and so a `\subsection{}` is level two (2). In your thesis it is likely that you will even use a `subsubsection{}`, which is level three (3). The depth to which the Table of Contents is formatted is set within `MastersDoctoralThesis.cls`. If you need this changed, you can do it in `main.tex`.

1.9. In Closing

You have reached the end of this mini-guide. You can now rename or overwrite this pdf file and begin writing your own `Chapter1.tex` and the rest of your thesis. The easy work of setting up the structure and framework has been taken care of for you. It's now your job to fill it out!

Good luck and have lots of fun!

Guide written by —
Sunil Patel: www.sunilpatel.co.uk
Vel: LaTeXTemplates.com

Capítulo 2

Introducción

2.1. Motivación

Las comunicaciones seguras nacen del deseo de protegernos: de proteger con quién nos comunicamos y el qué comunicamos.

De este deseo surgen multitud de protocolos de seguridad que hoy en día usamos sin darnos cuenta. Desde una simple consulta web hasta la felicitación de Año Nuevo, nuestras comunicaciones pasan por diversas operaciones para preservar su seguridad.

Esta seguridad viene generalmente proporcionada por la confianza que depositamos en ciertas organizaciones, las cuales crean una red de confianza sobre la que se sustenta todo este sistema. Pero, ¿qué sucede si no podemos confiar en estas entidades? ¿Y si queremos ser nosotros los responsables de proporcionar la seguridad?

Con esta idea comienza la búsqueda de una herramienta que permita a los usuarios ser los artífices de su propia red de confianza, el pilar central en seguridad.

2.2. Estructura de la memoria

Capítulo 3

Objetivos

3.1. Objetivo general

3.2. Objetivos específicos

3.3. Modelo de atacante

Capítulo 4

Estado del Arte

4.1. Java

Java es un lenguaje de programación de propósito general, orientado a objetos y concurrente. Originalmente fue desarrollado por James Gosling, Bill Joy y Guy Steele para Sun Microsystems en 1996 y fue adquirido por Oracle en 2010.

Fue diseñado para que los desarrolladores escribiesen una única vez su programa y pudieran ejecutarlo en cualquier máquina sin necesitar recompilarlo. Esto es posible debido a que las aplicaciones Java son compiladas a *bytecode* que luego es ejecutado en una **Java Virtual Machine (JVM)**, sin importar la arquitectura de la máquina. *(Gosling y col., 2015)*

FIGURA 4.1: Logo de Java
(Wikimedia, 2017)

4.2. Android

Android es un sistema operativo propiedad de Google, basado en el *kernel* de Linux. Está diseñado principalmente para dispositivos táctiles, como *smartphones* y *tablets*.

Android es compatible con las librerías de Java, por lo que la mayoría de las aplicaciones están escritas en ese lenguaje. Desde la versión 4.4 de Android, se utiliza **Android Runtime** (**ART**) como entorno de ejecución[1], el cual compila completamente el *bytecode* a código máquina durante la instalación de la aplicación. *(Wikipedia, 2017b)*

FIGURA 4.2: Logo de Android
(Wikimedia, 2014a)

4.3. Bouncy Castle

Bouncy Castle (BC) es una colección de APIs utilizados en criptografía. Estas APIs, entre otras cosas, proporcionan los siguientes servicios:

- Una API criptográfica *ligera* para Java y C#.
- Un proveedor para Java Cryptography Extension (JCE)[2] y para Java Cryptography Architecture (JCA).

Todas las APIs de BC están mantenidas por una organización caritativa australiana, conocida como **The Legion of the Bouncy Castle**. *(Bouncy Castle, 2013)*

FIGURA 4.3: The Legion of the Bouncy Castle, creadores de Bouncy Castle
(Bouncy Castle, 2013)

[1] En versiones anteriores se utilizaba Dalvik y su máquina virtual.
[2] JCE implementa encriptación, generación y protocolos de establecimiento de claves y algoritmos MAC.

4.4. Criptografía de clave simétrica

La *confidencialidad* en las comunicaciones es la causa principal por la cual surgen los algoritmos de encriptación. La **criptografía de clave simétrica** nos permite de manera segura comunicarnos con otra persona (o máquina) de manera que un tercero, aun teniendo el mensaje cifrado, no pudiera sacar nada de información de él.

Los algoritmos basados en este modelo utilizan un *secreto* compartido entre los dos puntos que se quieren comunicar. Una vez acordado, nadie que no posea este secreto podría descifrar ningún mensaje enviado por uno u otro extremo.

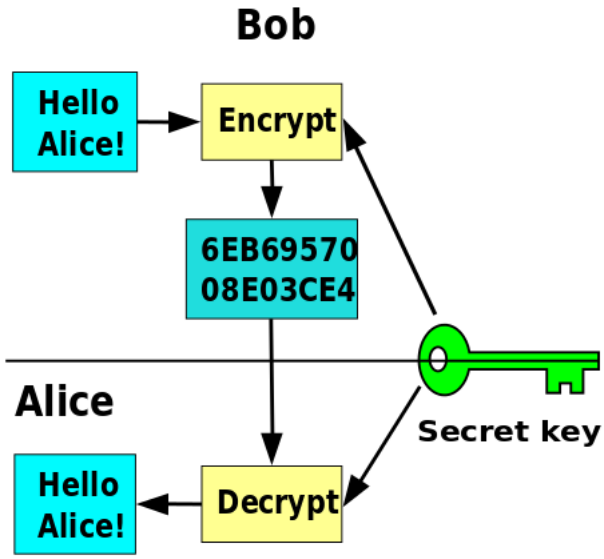

FIGURA 4.4: Esquema general de la criptografía de clave simétrica *(Wikimedia, 2014b)*

El problema de este modelo viene en el intercambio de las claves. Es necesario un canal seguro por el que comunicar las claves y este tipo de algoritmos no proveen ese servicio. *(Delfs y Knebl, 2007)*

Es por ello que normalmente se mezcla este tipo de criptografía con otro conocido como **criptografía de clave pública**, el cual veremos más adelante.

4.5. Cifrado por bloques

Dentro de la criptografía simétrica nos encontramos dos grandes algoritmos: los de tipo *stream ciphers* y los de tipo *block ciphers*. Nos centraremos en el segundo.

Un algoritmo basado en **cifrado por bloques** es aquel que, como su propio nombre indica, realiza un cifrado primero dividiendo el *plaintext* en bloques de un tamaño determinado[3] y luego cifrando por separado estos bloques, dando como resultado un *ciphertext*.

[3]Cuando el tamaño del fichero que queremos encriptar no es múltiplo del tamaño de bloque, el bloque final tendrá un tamaño diferente al resto. Esto se soluciona con técnicas de padding, lo cual se explica en el siguiente punto.

La mayoria de los *block ciphers* son iterativos, es decir, realizan la misma operación un determinado número de veces (*rounds*). Esta operación suele ser idéntica en todas las rondas, a excepción de la primera o la última, donde suele ser distinta.

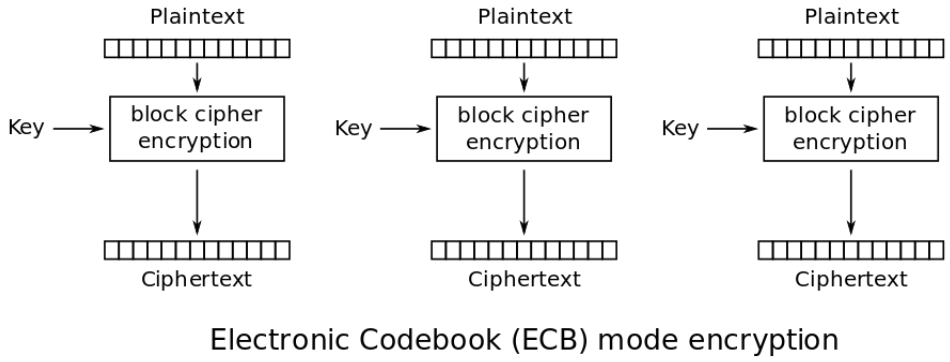

FIGURA 4.5: Cifrado usando el modo Electronic Codebook (ECB) *(Wikimedia, 2013c)*

Existen varios modos de operación para llevar a cabo un **cifrado por bloques**: ECB, CBC, OFB, CFB, etc. Cada uno de ellos opera los bloque de diferente forma, utilizando para ello técnicas muy diversas. *(Cusick y Stanica, 2009)*

4.6. Padding

Padding es el nombre que recibe la técnica que permite en criptografía por bloques expandir el último bloque del mensaje hasta lograr un tamaño deseado.

Es muy común que, en la criptografía por bloques, los fragmentos que encriptamos no tengan la itud que queremos para nuestro sistema. Para solucionar esto existen multitud de técnicas de **padding**, desde agregar al final del bloque un byte con un cierto valor, hasta simplemente rellenar con ceros.

El requisito indispensable que debe cumplir cualquier técnica de padding es que debe permitir al destinatario diferenciar los bytes del mensaje original de los byte de relleno. *(Balao, 2011)*

4.7. CBC

Cipher Block Chaining (CBC) es uno de los modos de operación para cifrado por bloques que hemos mencionado antes, y el que se ha decidido elegir para este proyecto.

Algunos modos como ECB (debido a como opera) tienen la característica de que, dada una clave determinada, cualquier *plaintext* siempre dará como resultado el mismo *ciphertext*. Si, como es nuestro caso, esta característica supone un problema, se opta por otros modos de operación como CBC, el cual soluciona esto. *(Wikipedia, 2017c)*

4.7.1. Modo de operación

CBC funciona combinando cada bloque de *plaintext* con el bloque de *ciphertext* justamente anterior. Obviamente, para el primer bloque no se dispone de un bloque cifrado anterior, por lo que se recurre al uso de un **vector de inicialización (IV)**. [4]

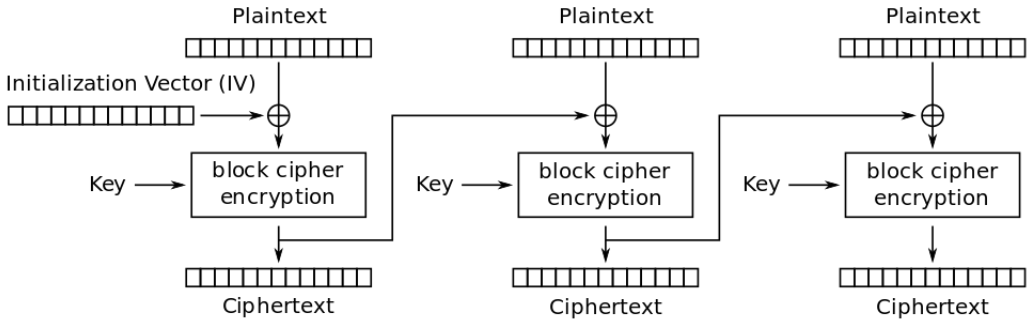

FIGURA 4.6: Cifrado usando el modo Cipher Block Chaining (CBC)
(Wikimedia, 2013b)

Como vemos en la Figura 4.6, en el cifrado con **CBC**, el primer bloque de texto y el IV son sometidos a una operación XOR. El resultado de esta operación se pasa por una función de cifrado[5], la cual nos dará el primer bloque cifrado. Este primer bloque es entonces pasado por un XOR junto con el segundo bloque de texto, y así sucesivamente.

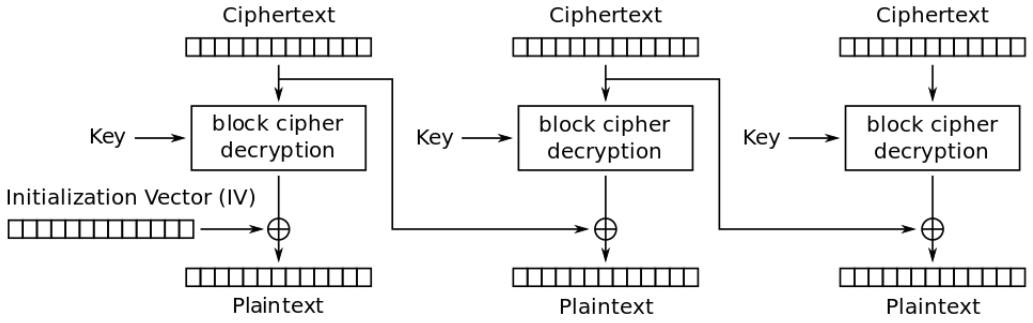

FIGURA 4.7: Descifrado usando Cipher Block Chaining (CBC)
(Wikimedia, 2013a)

A la hora de desencriptar, la función inversa a la utilizada en la encriptación es usada para obtener cada bloque de texto a partir de los bloques cifrados. Una representación de este proceso lo encontramos en la Figura 4.7.

[4] Un IV es un conjunto de bytes aleatorios que se usan para suplir la necesidad de un bloque cifrado anterior al primer bloque. Una muy mala práctica, por no decir prohibida, es la reutilización de un IV, ya que éste debe ser de uso único.

[5] La función de cifrado es la misma para todos los bloques.

El objetivo de un **cifrado por bloques** es poder paralelizar el proceso de encriptación. Lamentablemente, el cifrado con **CBC** debe ser secuencial, ya que para obtener cada bloque es necesario haber generado antes el anterior.

Sin embargo, el proceso de desencriptación con **CBC** sí que puede ser paralelizado, ya que las mútiples funciones de descifrado que se realizan no dependen de ningún bloque anterior. *(Dworkin, 2001)*

4.8. AES

Advanced Encryption Standard (AES), también conocido como **Rijndael** por sus creadores, es el resultado del proceso de búsqueda de un estándar para encriptación por parte del gobierno de los Estados Unidos.

AES es una variante de **Rijndael**, del cual solo toma algunos modos. Mientras que el algoritmo original puede tomar tamaños de bloque múltiplos de 32 bits,[6] **AES** únicamente opera con tamaños de bloque igual a 128 bits.

Con el tamaño de las claves ocurre algo parecido, ya que **AES** solo soporta tamaños de 128, 192 y 256 bits, mientras que el algoritmo original soporta unos cuantos más. *(Wikipedia, 2017a)*

4.8.1. State

Antes de meternos a hablar del algoritmo en sí, es necesario explicar una estructura que se usará constantemente.

El ***State*** es una estructura bidimensional de bytes con la que se operan los bytes de los bloques de entrada. Está compuesto por 4 filas y 4 columnas, dando un total de 32 celdas, en las cuales se almacenan los 16 bytes que forman un bloque. *(NIST, 2001)*

4.8.2. Transformaciones

El algoritmo consta de ciertas fases, una de ellas consiste en someter el *State* a unas cuantas rondas de transformaciones. Estas transformaciones son las siguientes:

- **SubBytes** – Mediante una transformación no lineal, los bytes del *State* son reemplazados por otros usando una tabla de sustitución. (Figura 4.8)
- **ShiftRows** – Los bytes de las 3 últimas filas del *State* se desplazan de manera cíclica, cada fila con un offset distinto. (Figura 4.9)
- **MixColumns** – Mediante una transformación lineal, las columnas del *State* se mezclan para producir unas nuevas. (Figura 4.10)
- **AddRoundKey** – En esta transformación, una *Round Key* se añade al *State* mediante una operación XOR. La itud de la *Round Key* debe ser igual al tamaño del *State*. (Figura 4.11)

(NIST, 2001)

[6] Con un mínimo de 128 bits y un máximo de 256.

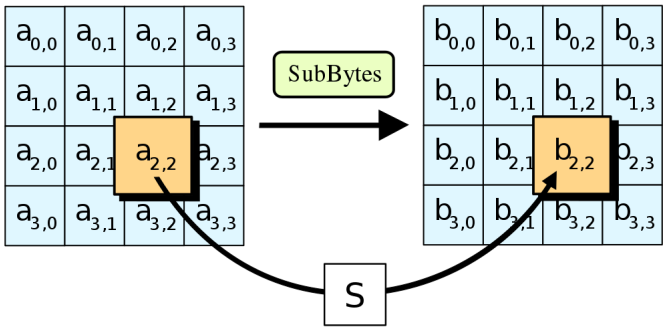

FIGURA 4.8: Operación SubBytes para AES
(Wikimedia, 2006d)

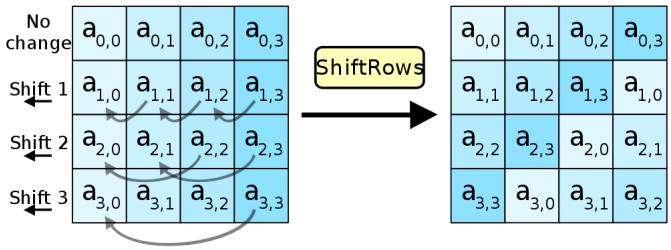

FIGURA 4.9: Operación ShiftRows para AES
(Wikimedia, 2006c)

4.8.3. Algoritmo

En **AES** el tamaño del bloque de entrada, del de salida y del *State* es de 128 bits, y el tamaño de la clave puede tomar los valores de 128, 192 ó 256 bits.

Para cada bloque de entrada, la primera etapa del algoritmo consiste en generar las *Round Keys* a partir de la clave, usando el esquema de claves Rijndael.[7]

Una vez conseguidas las *Round Keys*, se pasa por una ronda inicial especial en la cual solo se realiza la transformación *AddRoundKey*.

CUADRO 4.1: Combinaciones para el número de rondas en AES.

Key size (bits)	Block size (bits)	Rounds (Nr)
128	128	10
192	128	12
256	128	14

Después de esta etapa inicial, se pasa a las rondas habladas en el punto anterior. El número de rondas que lleva a cabo el algoritmo depende del tamaño de la clave, lo cual vemos representado en el Cuadro 4.1.

En todas estas rondas menos en la útlima, el orden de las transformaciones será siempre el mismo:

$$SubBytes \rightarrow ShiftRows \rightarrow MixColumns \rightarrow AddRoundKey$$

[7]Este esquema expande una clave en un número determinado de claves separadas.

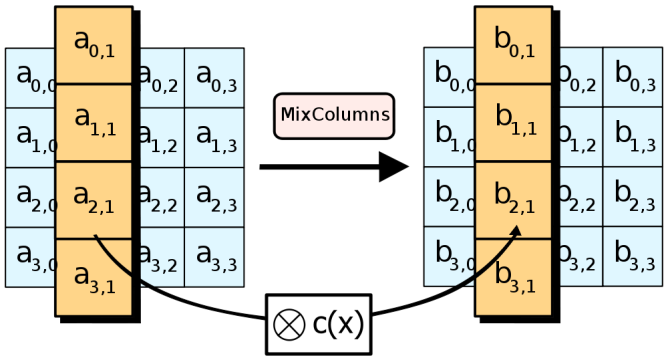

FIGURA 4.10: Operación MixColumns para AES
(Wikimedia, 2006b)

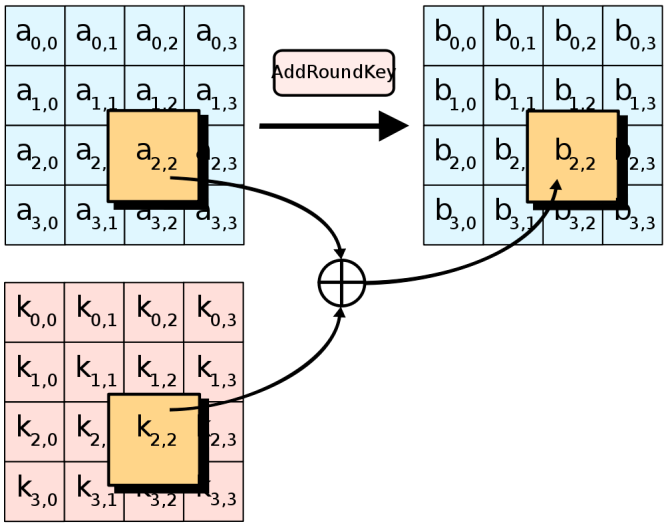

FIGURA 4.11: Operación AddRoundKey para AES
(Wikimedia, 2006a)

La última ronda es idéntica a las anteriores salvo por el hecho de que la transformación *MixColumns* no se realiza.

El resultado de todo esto será el *output* de nuestro bloque de entrada. *(NIST, 2001)*

4.9. Criptografía de clave pública

Como hemos visto, uno de los problemas que tiene la **criptografía de clave simétrica** es que necesita de un medio seguro para compartir la clave de cifrado entre los componentes de la comunicación. La **criptografía de clave pública** soluciona este problema al no ser necesario crear un canal seguro para el establecimiento de la clave, ya que para este modelo se hace uso de un par de claves: una *pública* y otra *privada*.

La esencia de este algoritmo radica en que un mensaje cifrado con una clave pública solo puede ser descifrado con su homóloga privada, y viceversa. De esta manera podemos, tal como su propio nombre indica, difundir públicamente nuestra clave *pública* mientras mantenemos oculta la *privada*.

Si lo que queremos es *confidencialidad* durante la comunicación, entonces encriptaremos el contenido del mensaje con la clave pública del destinatario, de forma que solo él podrá descifrarlo (Figura 4.12).

Por otra parte, también nos interesa que cuando alguien reciba nuestro mensaje pueda estar seguro de que realmente es nuestro. Para lograr esto, lo que hacemos es cifrar un resumen del mensaje (*hash*) con nuestra clave privada. De esta forma, cuando alguien lo descifre con nuestra clave pública y compruebe el resumen, podrá estar seguro de que proviene de nosotros.[8] Así conseguimos preservar la *integridad* y la *autentificación* del mensaje, dos parámetros muy importantes a tener en cuenta en seguridad. *(Hirsch, 2013)*

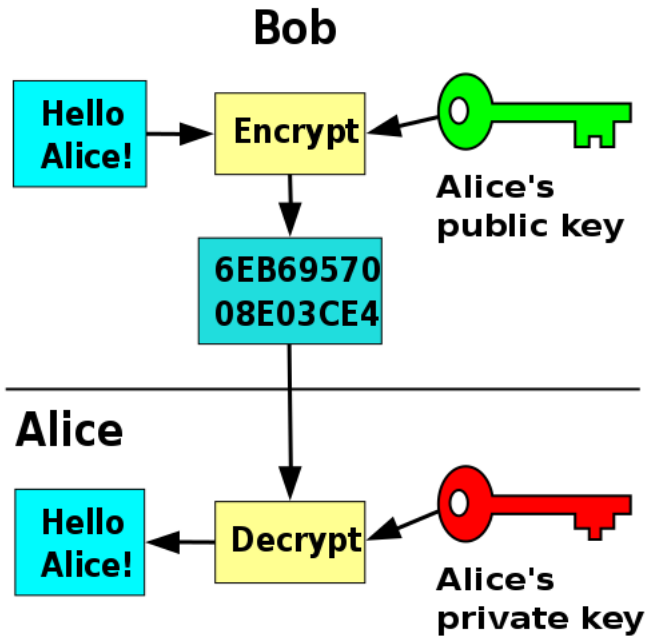

FIGURA 4.12: Esquema general del cifrado de clave pública *(Wikimedia, 2006e)*

4.10. RSA

Uno de los primeros algoritmos de criptografía pública (y el más utilizado hoy) es **RSA** (**Rivest–Shamir–Adleman**)[9]. Una de sus características más distintivas es el uso de números primos y la factorización de sus productos en la generación de las claves y en la encriptación/desencriptación de los mensajes (*factoring problem*).

RSA no es el más rápido de los algoritmos de criptografía que existen actualmente. Por ello, muchas veces se recurre a su uso en conjunto con otros algoritmos de criptografía simétrica, como **AES**. *(Wikipedia, 2017d)*

[8] Es, en esencia, el fundamento sobre el que se basa la firma digital.
[9] RSA debe su nombre a sus creadores: Ron Rivest, Adi Shamir y Leonard Adleman.

4.10.1. Clave pública RSA

Dentro del conjunto del par de claves **RSA**, la *pública* es la que se distribuye públicamente para que cualquiera que quisiera comunicarse con nosotros pudiera hacerlo de manera confidencial. Este tipo de claves consta de dos elementos:

- **n** – el módulo RSA, un entero positivo.
- **e** – el exponente público RSA, otro entero positivo.

Lo primero que se debe hacer para generar la clave *pública* es seleccionar de manera aleatoria dos números primos impares distintos, lo suficientemente grandes como para que sea computacionalmente inviable su descubrimiento por parte de un atacante. Una vez encontrados, el módulo **n** de la clave *pública* será el producto de estos dos números primos.[10]

$$n = p \cdot q$$

Ahora que tenemos el módulo de la clave definido, es hora de generar un exponente público **e** adecuado. Para ello, haremos uso de la función de Carmichael[11], la cual tiene la siguiente forma:

$$\lambda(n) = mcm^{12}(p-1, q-1)$$

El exponente público **e** será aquel número entero comprendido entre 3 y (**n** - 1) que satisfaga:

$$MCD^{13}(e, \lambda(n)) = 1$$

(Moriarty y col., 2016a)

4.10.2. Clave privada RSA

La clave *privada* es la que mantenemos oculta. Aunque existen varios modelos, generalmente consta de dos elementos:

- **n** – el módulo RSA, un entero positivo (Debe ser el mismo que el de la clave *pública*).
- **d** – el exponente privado RSA, un entero positivo.

Para calcular el exponente privado **d** deberemos elegir un entero positivo menor que **n**, que además cumpla:

$$e \cdot d \equiv 1 \pmod{\lambda(n)}$$

, donde **e** será el correspondiente exponente público. *(Moriarty y col., 2016b)*

4.10.3. Evaluación de la función RSA

Lo primero que debemos hacer es conseguir una representación de nuestro mensaje como un número entero, comprendido entre 0 y (**n** - 1).

[10] Por convención, se denotan como p y q.

[11] La función que se muestra solo es válida cuando n es el producto de dos números primos impares y distintos.

[12] Mínimo común múltiplo.

[13] Máximo común divisor.

Vamos a suponer que este mensaje queremos enviárselo a alguien cuya clave pública es (e, n). Si nuestro mensaje (recordemos que ahora es un entero) es M, para obtener el mensaje cifrado, aplicaremos la siguiente operación:

$$C \equiv E(M) \equiv M^e \pmod{n}$$

El entero C será del mismo tamaño que M y representará al mensaje cifrado.

Si ahora el destinatario quisiera descifrar el mensaje, solo tendría que revertir la operación con su clave privada (d, n) de la siguiente manera:

$$M \equiv D(C) \equiv C^d \pmod{n}$$

De esta manera, recuperaría el mensaje original.[14] *(Rivest, Shamir y Adleman, 1978)*

4.11. RSASSA-PSS

Antes hablamos de la necesidad de preservar la *integridad* y la *autentificación* en las comunicaciones. **RSASSA-PSS** es un algoritmo de firma digital que sirve precisamente para ello. Combina lo visto de RSA con un método de codificación llamado Probabilistic Signature Scheme (PSS). [15]

4.11.1. Probabilistic Signature Scheme (PSS)

PSS es un método de codificación[16] desarrollado por Mihir Bellare y Phillip Rogaway, los cuales buscaban mejorar los métodos que existían. Para ello, incluyeron en su esquema el uso de una *salt*,[17] lo cual haría más seguro el algoritmo frente a intentos de romperlo. *(Bellare y Rogaway, 1998)*

Como vemos en la Figura 4.13, **PSS** genera un resumen (*hash*) del mensaje. Este resumen se vuelve a pasar de nuevo por una función *hash*[18] junto a un *padding*[19] y la *salt* tal como muestra la figura.

El resultado de esta operación será la segunda de 3 piezas que conformarán nuestro mensaje codificado y la denotaremos como **H**. Para la primera (*maskedDB*), generaremos una máscara[20] de **H** y haremos una operación *xor* con ella y la salt (junto a otro *padding*).[21]

Ahora solo nos quedará añadir al final de nuestra salida un byte con valor *0xbc* y ya habremos acabado. *(Moriarty y col., 2016c)*

[14] Para cifrar con una clave privada y descifrar con una pública se hacen las mismas operaciones, cambiando e y d donde corresponda.

[15] Las siglas SSA corresponden a Signature Scheme with Appendix, lo que quiere decir que la firma va añadida al final del mensaje o junto a él.

[16] Un método de codificación es una operación que permite convertir un carácter de un determinado conjunto en un símbolo de otro sistema de representación.

[17] Bits aleatorios.

[18] Las dos funciones *hash* que se usan en el esquema deben ser la misma.

[19] El primer *padding* estará formado por 8 bytes con valor *0x00*.

[20] Una máscara es muy parecida a una *hash*. Mientras la segunda tiene un tamaño determinado, una máscara puede tomar distintos tamaños según la necesidad.

[21] Este *padding* está formado por una cantidad variable de bytes con valor *0x00*, teniendo al final un byte de valor *0x01*.

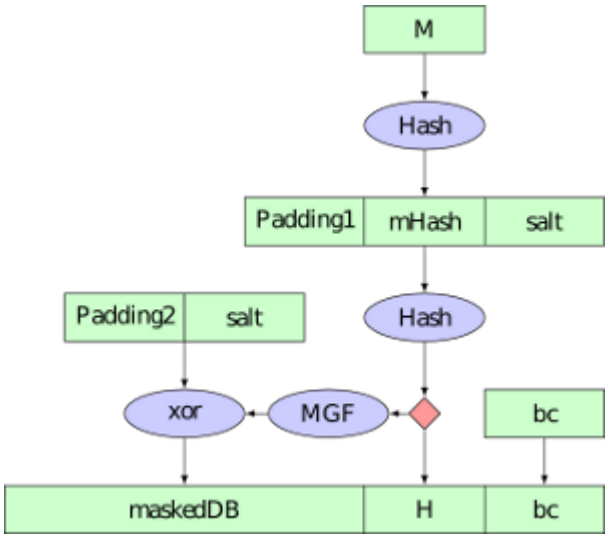

FIGURA 4.13: PSS de acuerdo a PKCS #1 V2.2 / RFC 8017
(Böck, 2011)

4.12. HTTP

Hypertext Transfer Protocol (**HTTP**) es un protocolo con la ligereza y la velocidad necesarias para un sistema de información hipermedia[22] colaborativo y distribuido.

Es un protocolo genérico orientado a objetos sin estado, que se puede utilizar para muchas tareas similares, como servidores de nombres y sistemas distribuidos orientados a objetos mediante la ampliación de los comandos (métodos) utilizados.

Una característica de **HTTP** es la negociación de representación de datos, lo que permite que los sistemas se construyan independientemente del desarrollo de nuevas representaciones avanzadas. *(W3C, 1992)*

[22]El término hipermedia engloba toda aquella información que incluye imagen, audio, vídeo, texto, etc.

Capítulo 5

Diseño e implementación

5.1. Arquitectura de seguridad

5.2. Arquitectura del software

Capítulo 6

Resultados

6.1. Ejemplos de utilidad

6.2. Problemas encontrados

Capítulo 7

Conclusiones finales

7.1. Objetivos alcanzados

7.2. Líneas futuras

Apéndice A

Frequently Asked Questions

A.1. How do I change the colors of links?

The color of links can be changed to your liking using:
`\hypersetup{urlcolor=red}`, or
`\hypersetup{citecolor=green}`, or
`\hypersetup{allcolor=blue}`.
If you want to completely hide the links, you can use:
`\hypersetup{allcolors=.}`, or even better:
`\hypersetup{hidelinks}`.
If you want to have obvious links in the PDF but not the printed text, use:
`\hypersetup{colorlinks=false}`.

Bibliografía

Balao, Martín (2011). *Criptografía: Padding + ECB + CBC*. URL: http://martin.com.uy/sec/criptografia-padding-ecb-cbc/.

Bellare, Mihir y Phillip Rogaway (1998). *PSS: Provably Secure Encoding Method for Digital Signatures*. URL: http://grouper.ieee.org/groups/1363/P1363a/contributions/pss-submission.pdf.

Bouncy Castle, The Legion of the (2013). *BC Home*. URL: http://bouncycastle.org/.

Böck, Johannes (2011). *RSA-PSS – Provable secure RSA Signatures and their Implementation*. URL: https://rsapss.hboeck.de/rsapss.pdf.

Cusick, Thomas W. y Pantelimon Stanica (2009). «Cryptographic Boolean functions and applications». En: ed. por Academic Press. Cap. 7, págs. 158-159.

Delfs, Hans y Helmut Knebl (2007). *Introduction to cryptography: principles and applications*. Ed. por Ueli Maurer. Springer.

Dworkin, Morris (2001). *Recommendation for Block Cipher Modes of Operation*. Inf. téc. 800-38A. NIST. URL: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf.

Gosling, James y col. (2015). *The Java® Language Specification*. URL: https://docs.oracle.com/javase/specs/jls/se8/jls8.pdf.

Hirsch, Frederick J. (2013). *SSL/TLS Strong Encryption: An Introduction*. URL: http://httpd.apache.org/docs/2.2/ssl/ssl_intro.html#cryptographictech.

Moriarty, Ed. K. y col. (2016a). *PKCS #1: RSA Cryptography Specifications Version 2.2*. Inf. téc. RFC 8017. IETF, págs. 8-9. URL: https://tools.ietf.org/html/rfc8017#section-3.1.

— (2016b). *PKCS #1: RSA Cryptography Specifications Version 2.2*. Inf. téc. RFC 8017. IETF, págs. 9-10. URL: https://tools.ietf.org/html/rfc8017#section-3.2.

— (2016c). *PKCS #1: RSA Cryptography Specifications Version 2.2*. Inf. téc. RFC 8017. IETF, págs. 42-43. URL: https://tools.ietf.org/html/rfc8017#section-9.1.1.

NIST (2001). *Announcing the ADVANCED ENCRYPTION STANDARD (AES)*. Inf. téc. FIPS 197. NIST. URL: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf.

Rivest, R. L., A. Shamir y L. Adleman (1978). *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*. URL: http://people.csail.mit.edu/rivest/Rsapaper.pdf.

W3C (1992). *Basic HTTP as defined in 1992*. URL: https://www.w3.org/Protocols/HTTP/HTTP2.html.

Wikimedia (2006a). *File:AES-AddRoundKey.svg*. URL: https://commons.wikimedia.org/wiki/File:AES-AddRoundKey.svg.

— (2006b). *File:AES-MixColumns.svg*. URL: https://commons.wikimedia.org/wiki/File:AES-MixColumns.svg.

— (2006c). *File:AES-ShiftRows.svg*. URL: https://commons.wikimedia.org/wiki/File:AES-ShiftRows.svg.

— (2006d). *File:AES-SubBytes.svg*. URL: https://commons.wikimedia.org/wiki/File:AES-SubBytes.svg.

Wikimedia (2006e). *File:Public key encryption.svg*. URL: https://commons.wikimedia.org/wiki/File:Public_key_encryption.svg.
— (2013a). *File:CBC decryption.svg*. URL: https://commons.wikimedia.org/wiki/File:CBC_decryption.svg.
— (2013b). *File:CBC encryption.svg*. URL: https://commons.wikimedia.org/wiki/File:CBC_encryption.svg.
— (2013c). *File:ECB encryption.svg*. URL: https://commons.wikimedia.org/wiki/File:ECB_encryption.svg.
— (2014a). *File:Android robot 2014.svg*. URL: https://commons.wikimedia.org/wiki/File:Android_robot_2014.svg.
— (2014b). *File:Symmetric key encryption.svg*. URL: https://commons.wikimedia.org/wiki/File:Symmetric_key_encryption.svg.
— (2017). *File:Java programming language logo.svg*. URL: https://en.wikipedia.org/wiki/File:Java_programming_language_logo.svg.
Wikipedia (2017a). *Advanced Encryption Standard*. URL: https://en.wikipedia.org/w/index.php?title=Advanced_Encryption_Standard&oldid=809193008.
— (2017b). *Android (operating system)*. URL: https://en.wikipedia.org/w/index.php?title=Android_(operating_system)&oldid=805639596.
— (2017c). *Block cipher mode of operation*. URL: https://en.wikipedia.org/w/index.php?title=Block_cipher_mode_of_operation&oldid=809021817#Common_modes.
— (2017d). *RSA (cryptosystem)*. URL: https://en.wikipedia.org/w/index.php?title=RSA_(cryptosystem)&oldid=806096352.